Deep Anomaly Detection in Hyperspectral Images Based on Membership Maps and Object Area Filtering

Mahdi Yousefan, Hamid Esmaeili Najafabadi, *Member, IEEE,* Hossein Amirkhani,
Henry Leung, *Fellow, IEEE,* Vahid Hajihashemi,

***Abstract*—In this paper, we propose a novel framework for transferred deep learning-based anomaly detection in hyperspectral images. The proposed framework includes five main steps. First, the image's spectral dimension is reduced by applying the principal component analysis (PCA). A convolutional neural network (CNN)-like deep network is then trained using only one image to learn the pixels' similarities in a picture. This learned is applied to extract objects in the image using connected component filtering. Next, irrelevant objects are removed by comparing their area to an acceptable anomaly area range. The final result is obtained by multiplying the network output and binary map of anomalies. Extensive experimental evaluations demonstrate that the proposed framework substantially outperforms a significant number of comparable state-of-the-art methods. Finally, we empirically verify that the deep network exhibits excellent domain adaptability.**

***Index Terms*—Anomaly detection, convolutional neural network, hyperspectral image, object area filtering**

I. Introduction

HYPERSPECTRAL narrow-band imaging has emerged as a practical solution to remote sensing applications. It allows the distinction of different materials due to its high spectral resolution [1, 2]. Primarily, two types of applications employ their hyperspectral features: anomaly detection and anomaly classification [3, 4]. In anomaly detection, targets in hyperspectral images are often physical objects with some unique spectral characteristics. These targets may include particular species in agriculture and ecology, rare animals in geology, a ship in a sea background, or planes in an airport [5–7]. Almost all state-of-the-art approaches include at least a learning subroutine. In this regard, anomaly detection and anomaly classification can be categorized into supervised and unsupervised based on its learning, where the latter often employs statistical methods. Reed-Xiaoli (RX) is an instance of unsupervised algorithms that employ the sample correlation matrix to detect anomalies. This method can use a full hyperspectral image, known as global RX (GRX) [8], or a single local window around each pixel, known as local RX (LRX) [9]. Reference [10] proposes a nonlinear version of the RX. It extends the RX to a feature space associated with the original input space via a specific nonlinear map named kernel RX (KRX). Generally, the KRX can be implemented by adapting the RX algorithm to the feature space by utilizing the kernels that implicitly compute dot products. This method is proven to yield high performance in anomaly detection and change detection. A variant of KRX is presented in [11] as cluster KRX (CKRX), which becomes KRX under certain conditions. This method's main idea is to group background pixels into clusters and then apply a fast eigendecomposition to generate an anomaly detection index. Reference [12] develops an anomaly detection method for hyperspectral imagery (HSI) based on low-rank representation, incorporating a spatial constraint. A hyperspectral anomaly detection algorithm is investigated in [13] for real-time applications with push-broom sensors. Reference [14] studies an anomaly detection for HSI based on the regularized subspace method and collaborative representation, while the latter is also studied in [15]. An HSI anomaly detection model is proposed using background joint sparse representation (BJSR) in [16]. Reference [17] introduces a background anomaly component projection and separation optimized filter (BASO) for anomaly detection in hyperspectral images. In [18] and [19], two anomaly detection HSI methods are presented based on tensor decomposition and sparsity score estimation, respectively.

Recently, attribute edge-preserving (AED) filtering is proposed to detect targets in hyperspectral images [20]. In this method, areas with particular spectral features are detected through attribute filtering and Boolean map-based fusion to obtain an initial pixel-wise detection, which is then refined through edge-preserving. Reference [21] generalizes the AED using multiscale attributing. It is then employed to generate multiscale anomaly detection maps. In [22], a differential attribute profile anomaly detection (DAPAD) method utilizes principal component analysis (PCA) and differential attribute profiling (DAP) to extract spectral and spatial features from HSI, respectively.

Meanwhile, convolutional neural networks (CNN) are employed in HSI analysis due to their powerful feature extraction abilities. In [23], hyperspectral target detection is presented

Mahdi Yousefan is currently with the Department of Electrical and Computer Engineering, Yazd University, Yazd, Iran. Email: m.yousefan1998@gmail.com. Phone: +989391541146.

Hamid Esmaeili Najafabadi is currently a PostDoc associate at the Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada. Email: hamid.esmaeili@gmail.com. Phone: +1(587) 322-3311.

H. Amirkhani is with the Technology and Engineering Department, University of Qom, Qom 37181-17469, Iran. Email: amirkhani@qom.ac.ir.

Prof. Henry Leung is with the Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada. Email: leungh@ucalgary.ca.

Vahid Hajihashemi was with the Faculty of Engineering, Kharazmi University, Tehran, Iran. Email: Hajihashemi.vahid@ieee.org.

using pixel pair features (PPFs) and spatial PPFs (SPPFs). The CNNs are also applied to HSI classification in addition to target detection [24–28]. Reference [29] develops a deep CNN detection (CNND) algorithm to extract the hyperspectral images' spatial features in several steps.

Although it improves the performance by applying a deep CNN, the CNND requires a considerable amount of time compared to other methods. On the other hand, the data in HSI pictures are mostly redundant. By removing some of this redundancy through pre-processing, a less deep CNN can be applied to learn the similarities. Furthermore, none of the mentioned methods can present both short computation time and superior performance at the same time.

In this paper, we apply deep CNN for anomaly detection. Different from the literature, our contributions can be summarized as:

- A characteristic image is extracted from the HSI using principal component analysis (PCA). It considerably shrinks the amount of data needed to be processed. The pixel pairs are then fed into a novel network to obtain the similarity between a generic pixel and its neighbors. The designed network can be considered as a two-stage CNN-like deep network. The first stage is constituted from a series of convolution layers, followed by a Dropout layer. After a flattening layer, the data is fed into several dense layers, followed by a dropout layer. Finally, a dense layer with sigmoid activation is applied to get the detection probabilities. Compared to CNND, the introduced architecture has more dense layers and is not as deep, which results in less computational complexity and time.
- An object area filtering (OAF) is applied to improve performance. The output of the OAF is a binary image whose undesired objects are removed. Finally, the result is obtained by multiplying the network and OAF outputs. Using the PCA and OAF methods, we detect the targets in less time while presenting more accuracy than several state-of-the-art methods.

This paper is organized as follows. The next Section is devoted to describing the pre-processing applied in the proposed approach. In Section III, the details of the proposed method are elaborated. It is composed of several subsections, each detailing an essential part of the proposed method. We evaluate the newly-proposed method on a practical dataset in Section IV. The scores are then compared to half a dozen state-of-the-art algorithms. Finally, the conclusion is given in Section V.

II. Pre-Processing

A hyperspectral image is a 3-D tensor of dimensions $H \times W \times C$, where H, W, and C represent the height, weight, and channels or spectral bands, respectively. We consider a pixel-wise learning process. That is, we deal with a $1 \times 1 \times D$ tensors of data in the training process, where D might vary between the layers of the network.

The HSI often consists of a significant amount of data in a single image. Although a deep CNN can reduce the data in the first several layers, using a very deep CNN is not an efficient way to reduce the dimensionality due to two reasons. First, the convolution layers need to learn, which is a very time-consuming process. Second, the correlation among different layers in hyperspectral images is high. This fact can be exploited to reduce dimensionality without losing much information. Here, we first employ PCA as a pre-processing to reduce the dimensionality of the HSI efficiently. In this light, the channels of the hyperspectral image are reduced to $D < C$. We process each pixel independently in the learning subroutine. Therefore, the PCA changes the network input from $1 \times 1 \times C$ to $1 \times 1 \times D$.

III. Proposed Approach

Generally, the proposed framework can be broken down to five main steps:

A. Designing a novel well-designed network, generating the training dataset from a single image, and employing it to train the network
B. Considering a generic pixel, calculating and averaging the similarities between the generic pixel and its neighbors using the trained network
C. Applying object area filtering to remove irrelevant objects
D. Generating the output by multiplying the result of the steps (C) and (D)

The details of each step are elaborated in bellow:

A. Network Architecture and Training Process

After dimensionality reduction, the data is fed into the network. The network essentially learns the similarities between two pixels, in the sense that the two are similar if they belong to the same class in the picture. The output of the network is the probability of the two pixels belonging to the same object. Fig. 1 briefly depicts the architecture of the network. The designed network consists of 11 layers to classify a pixel pair into two classes: similar or dissimilar. The first layer of the proposed network is a convolutional layer with 30 filters. Afterward, a convolutional layer with 60 trainable filters is used as the second layer. The third and fourth layers are convolutional layers with 30 and 10 filters, respectively. After convolutional layers, a dropout layer is employed to reduce overfitting. The sixth layer is a flattening layer which flattens the output of the previous layer. Consequently, three hidden fully-connected layers are applied with 30, 20, and 10 hidden units, respectively. In the third dense layer, a L_1 regularizer with a coefficient 0.01 is used to achieve better generalization. The next layer is a dropout layer. The network architecture concludes in a fully connected layer with a sigmoid activation function.

Interestingly, the training process involves only one picture with known regions in it (see also [29, 30]). In particular, Salinas scene [31] with 16 known regions is employed here. The training data consist of two types of pixel pairs. The first is similarity data denoted as Category 1, where all pixel pairs belong to the same class in the picture. Category 2 is a dissimilarity set, consisting of pixel pairs from different classes. An adequate number of training data is fed into the

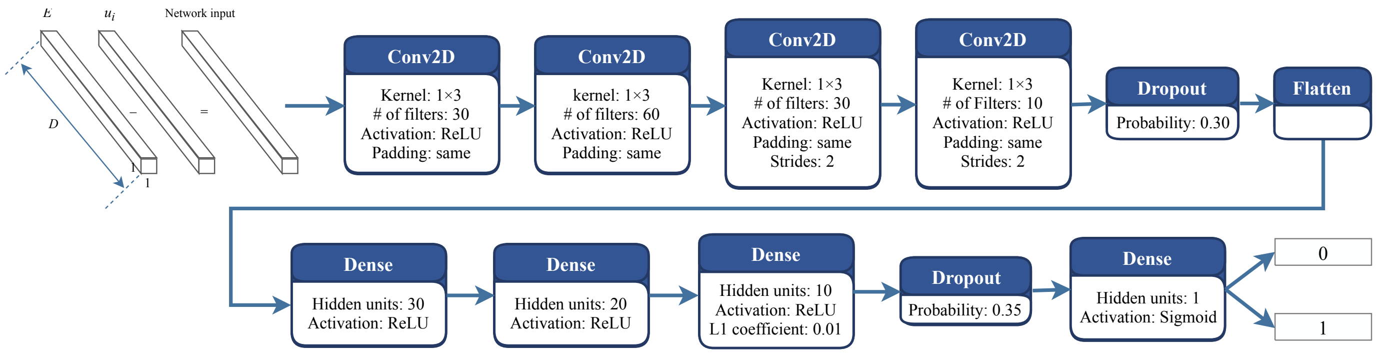

Fig. 1: The architecture of the proposed CNN. The network essentially learns the pixel-pair differences to determine if the two input pixels are from the same class

network to train the network. Furthermore, the binary cross-entropy map is used as the loss function, where stochastic gradient descent (SGD) is employed to minimize it. The parameters of the applied optimizer and network metrics are explained in Section III.

B. Calculating and Averaging the Similarities Between Test Pixel and Its Neighbors Through the Trained Network

The similarity score between two pixels can be employed to determine the objects in a picture. In this step, we run through the picture by a dual window structure given in Fig. 2, where the center pixel is denoted by E. We use the notation (w_{in}, w_{out}) to designate a windowing structure where the inner and outer windows have odd edge lengths w_{in} and w_{out}, respectively. For instance, $(3, 7)$ represents a square of length 3 surrounded by a square of length 7. Here, the goal is to compute the average similarity score between the center pixel, E, and the pixels between the two square windows, known as peripheral pixels. Obviously, the number of E peripheral pixels is calculated as below:

$$N = (w_{out}^2 - w_{in}^2). \tag{1}$$

Consequently, the similarity score between the generic pixel E and each peripheral pixel is calculated using the trained network and gathered into the vector $\mathbf{s}_E = [s_1, s_2, \cdots, s_N]^T$. The final score is generated by averaging the similarity score over peripheral pixels:

$$M(E) = \frac{1}{N}\sum_{i=1}^{N} s_i. \tag{2}$$

This procedure is performed for each pixel in the image using the same window. Here, $M(E)$ is a real number between zero and one, which can be interpreted as the membership of the pixel, E, to its corresponding object. That is, the pixels which are in the center of an object get higher values compared to others. We call the resulting image as the membership map and denote it with M.

C. Object Area Filtering

Here, we desire to find connected components and calculate their areas. To this end, a binary image is required instead of a membership map. Therefore, thresholding is applied to the membership map to get a binary map of objects. That is,

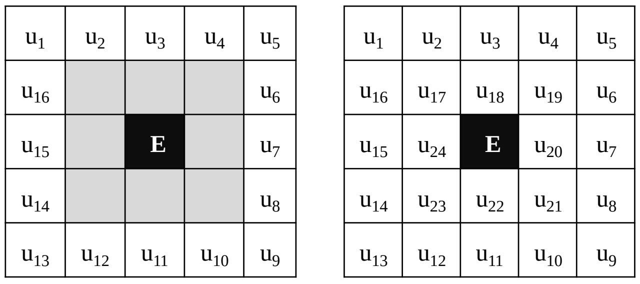
u_1 u_2 u_3 u_4 u_5
u_{16} u_6
u_{15} E u_7
u_{14} u_8
u_{13} u_{12} u_{11} u_{10} u_9
u_1 u_2 u_3 u_4 u_5
u_{16} u_{17} u_{18} u_{19} u_6
u_{15} u_{24} E u_{20} u_7
u_{14} u_{23} u_{22} u_{21} u_8
u_{13} u_{12} u_{11} u_{10} u_9

Fig. 2: Two different window size configurations: the left depicts a $(w_{in}, w_{out}) = (3, 5)$ window while right shows a $(1, 5)$ window, u_1 to u_N are the peripheral pixels which their similarity with E is of interest

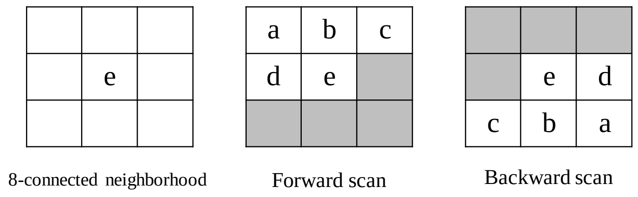

Fig. 3: Three different masks and neighborhood of pixel e used in OAF algorithm

$M(E)$ is compared with a predefined threshold T. If $M(E)$ is bigger than T, the pixel E is more likely an anomaly pixel; otherwise, it is a background pixel. Then, object area filtering (OAF) is employed to remove those objects which are not anomalies. This two-pass algorithm is similar to the connected component filtering given in [32]. Fig. 3 shows the 8-connected neighborhood as well as backward and forward scan masks used in this algorithm.

This algorithm employs a union-find data structure (UFDS) to record equivalence information among the provisional labels after the first scan. A UFDS can be viewed conceptually as a rooted tree, where each node of a tree is a provisional label, and each edge represents the equivalence between two labels.

The two passes of UFDS are:

1) First pass: The objects in the image are found. The current pixel is e, and a, b, c, and d are its neighbors. Let 1 and 0 be represented as white and black, respectively. Black pixels are part of the background and are ignored during scanning.

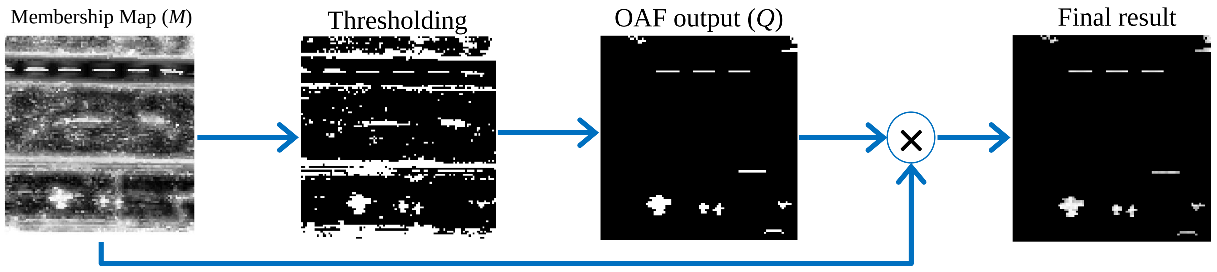

Fig. 4: A simplified schematic summarizing the procedures in the proposed anomaly detection method

If a pixel lies outside the bounds of the image, it is white by default. Therefore, there are six states in forwarding scan mask:

1. If the current pixel is white, it is not a component. Therefore, we ignore it.
2. If pixel b is in the image and black, a, d and c are its neighbors, and they are all part of the same component. Therefore, it is sufficient to assign b's label to e.
3. If pixel c is in the image and black, b is its neighbor, but a and d are not, then:
   - If pixel a is in the image and black, a and c are connected through e. Therefore, the Union is applied to their sets.
   - If pixel d is in the image and black, then d and c are connected through e. Therefore, the Union is applied to their sets.
4. If pixel a is in the image and black, we already know b, and c are white, d is a's neighbor, so they already have the same label. Therefore, simply assign a's label to e.
5. If pixel d is in the image and black, we already know a, b and c are white, therefore we assign d's label to e.
6. Otherwise, all the neighboring pixels are white, so the current pixel is a new component.

2) Second pass: We classify connected components based on their areas. The anomalies area is within a specific range, known as the acceptable anomaly area range (AARR). In this regard, the objects are removed if their size is not in this range of anomalies. By this process, the anomalies are filtered. The result of this step is a binary map denoted by Q.

D. Multiplication of Membership Map and OAF Output

At last, the output is obtained by multiplying the previous result (denoted as Q) and the membership map

$$result = Q \times M, \tag{3}$$

where the multiplication is pixel-wise. Fig. 4 depicts the output of each steps in process of the proposed anomaly detection algorithm.

IV. Numerical Experimentation

In this Section, we evaluate the proposed method in terms of performance and computation time. In this regard, the proposed approach is compared with six different counterpart algorithms. In particular, the CNND [29], BASO [17], LAD [33], RX [8], LRX [9], and CKRX [11] are chosen as benchmarks algorithms, where all are very recent algorithms detecting anomaly objects in HSI or established traditional benchmarks. All simulations are performed using Matlab 2016 on a Core-i5 PC with 8 Gbytes of RAM unless expressed explicitly. Python 3.6 and TensorFlow[1] are used for the deep learning algorithms, where all processing are performed on CPU and not GPU. In particular, BASO, LAD, RX, LRX, and CKRX algorithms are implemented in MATLAB, which is the original language they are proposed for. The proposed approach and CNND are implemented using Keras of the TensorFlow library in Python.

For all evaluations, Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) dataset[2] is used [31].

A. Train and Test Data Sets

We use the Salinas scene[3] from the AVIRIS dataset for the training purposes, which was captured by the 224-band Airborne AVIRIS sensor over Salinas Valley, California, and is characterized by high spatial resolution (3.7 meters/pixels). The image contains 512 lines by 217 samples and includes vegetables, bare soils, and vineyard fields. The train image ground truth has 16 categories (see [30] for details). Accordingly, the pixel pairs of the same (Category 1) and different (Category 2) objects are constructed, and the network is trained by them.

Meanwhile, four test images are also collected from AVIRIS, which are depicted in Fig. 5. All images are of the size 100×100 except for image (c) which is 150×150.

B. Parameter Setup

In addition to neural network weights, there are some hyperparameters which should be tuned during the training. The first is D, which shows the number of selected features by applying PCA. It is worth noting that lower D means less spectral features and computation time and vice versa. We observed that $D = 10$ is a suitable value in the simulations. The impact of D on overall performance is investigated in a supporting document to this paper [30].

[1]https://www.tensorflow.org/guide/keras
[2]http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes
[3]This image can be reached at: https://rslab.ut.ac.ir/data

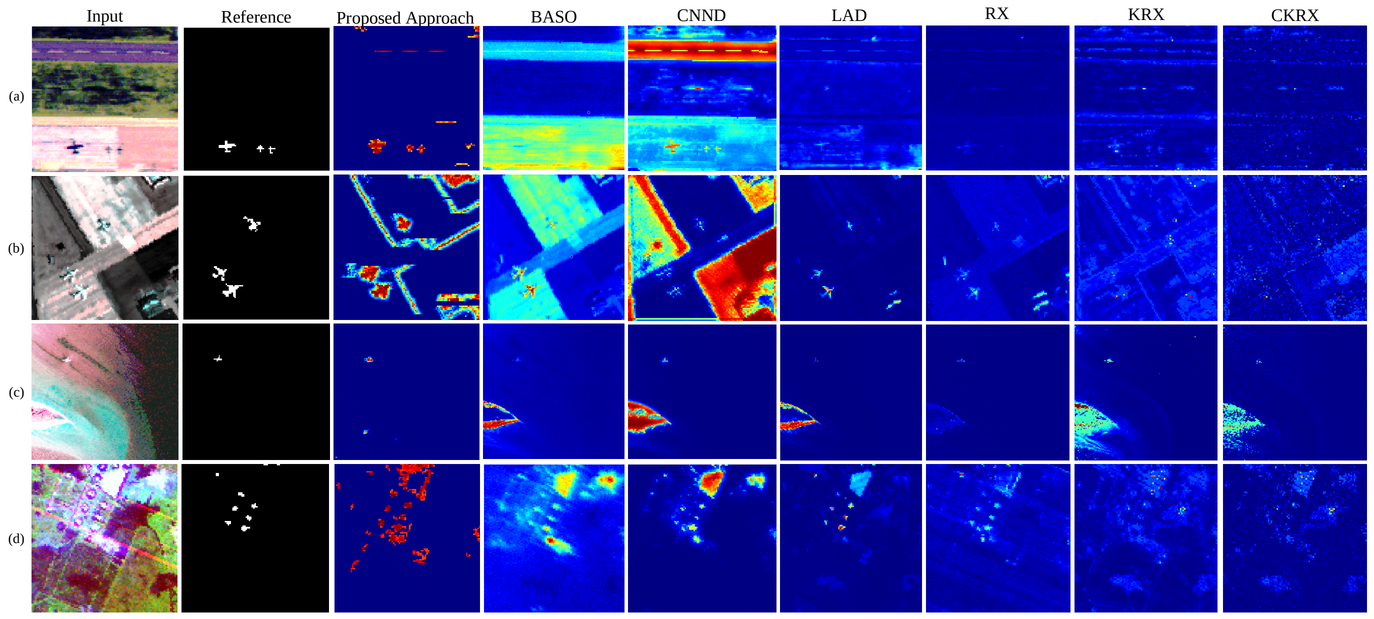

Fig. 5: The first column is color composites of test HSIs, while second column depicts ground truth and other columns are result of different detectors

(a)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	12	0.9878	0.53	[13, 19] U [35, 40]
(3, 5)	16	0.9960	0.55	[6, 20] U [65, 70]
(5, 7)	20	0.9959	0.65	[13, 23] U [45, 770]
(3, 7)	30	**0.9972**	0.65	[9, 18] U [50, 70]

(b)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	15	0.9736	0.01	All
(3, 5)	19	**0.9857**	0.10	$[120, +\infty)$
(5, 7)	21	0.9828	0.40	$[15, +\infty)$
(3, 7)	41	0.9840	0.60	$[55, +\infty)$
(1, 7)	39	0.9851	0.10	$[55, +\infty)$

(c)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	27	**0.9998**	0.01	[2, 60]
(3, 5)	36	0.9473	0.20	[20, 35]
(5, 7)	42	0.8682	0.25	[8, 30]
(3, 7)	75	0.8684	0.35	[15, 20]

(d)

(w_{in}, w_{out})	Time (sec)	AUC	Threshold	AARR (pixels)
(1, 3)	12	0.9873	0.50	$[5, +\infty)$
(3, 5)	15	0.9894	0.65	$[5, +\infty)$
(5, 7)	22	0.9942	0.70	$[4, +\infty)$
(3, 7)	32	0.9927	0.60	All
(1, 12)	110	**0.9954**	0.80	$[4, +\infty)$

TABLE I: Comparison of the performance measures of the proposed approach for images (a), (b), (c) and (d), when different window sizes and AARR intervals are applied

(a)

Image	Proposed Approach	BASO	CNND	LAD	RX	KRX	CKRX
(a)	**0.9972**	0.7292	0.8546	0.8076	0.9526	0.9516	0.5310
(b)	**0.9857**	0.8840	0.3849	0.9855	0.9403	0.8681	0.3722
(c)	**0.9998**	0.9589	0.9571	0.9203	0.9807	0.9293	0.8168
(d)	**0.9954**	0.9376	0.9521	0.9853	0.9907	0.9282	0.8180

(b)

Image	Proposed Approach	BASO	CNND	LAD	RX	KRX	CKRX
(a)	30	16	1505	**0.19**	0.26	2007.2	247.9
(b)	18.9	18.3	107	**0.21**	0.22	1739.6	275.6
(c)	27	73	1203	**0.39**	0.66	7854	465.6
(d)	110	32	807	**0.19**	0.26	2080	264.2

TABLE II: Comparison of AUC (a) performance and (b) computation time among the proposed and benchmark approaches

1) Training: One of the hyperparameters in the training phase is the learning rate, which is a configurable hyperparameter used in the neural networks' training and often is chosen between 0.0 and 1.0. It controls how a neural network takes steps on the estimated direction toward the minimum. Here, the proper learning rate is 0.01.

After each update, we apply weight decay to prevent the weights from growing too large. Our value of the weight decay is 10^{-6}. The next one is momentum, a hyperparameter between 0 and 1 that increases the size of the steps taken towards the minimum by trying to avoid zig-zagging around the axis perpendicular to the path toward to minimum point. The accepted value for the designed network momentum is 0.9.

2) Prediction: There are three parameters in the prediction phase: window size, a threshold to obtain the binary image, and an accepted range for connected component objects.

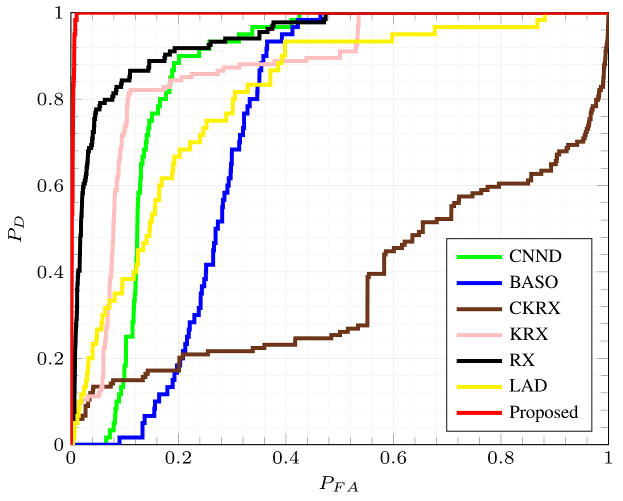

Fig. 6: ROC comparisons among the proposed and benchmark methods applied to image (a)

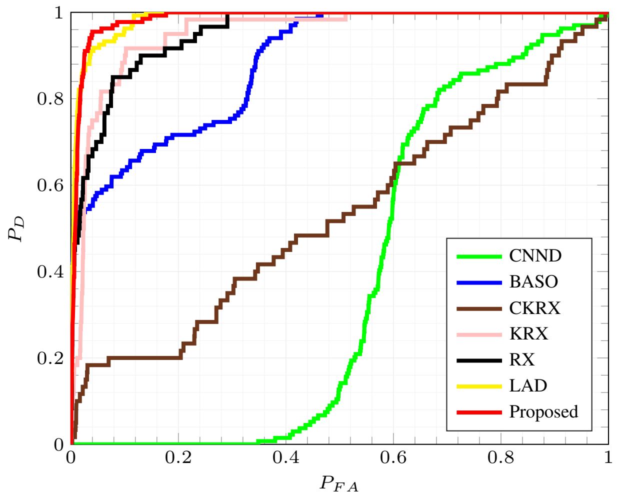

Fig. 7: ROC comparisons among the proposed and benchmark methods applied to image (b)

There are two types of window sizes, which are the single window and dual window. In the single window, the center pixel's subtractions and its neighbors are fed into the network. In the dual window, the network's input is the subtraction of the center pixel and the pixels between the inner window and exterior window. The effect of the window size is illustrated in Table I.

Meanwhile, it is imperative to choose the best value for the threshold to obtain the binary image from the OAF's network output because it influences the final result. In our experiments, each image has a different binary threshold, which is shown in Table I. The last hyperparameter in the prediction phase is the accepted range for objects found as anomalies. Accordingly, the AARR is written in Table I for different images.

C. Domain Adaptation

The applicability of the proposed method to the data gathered from another source is called domain adaptation. To evaluate this feature of the proposed approach, we train the network with a new image collected by another sensor. Then,

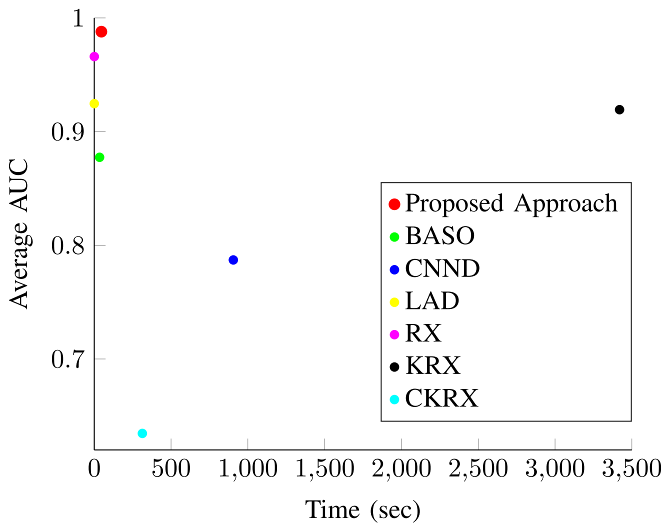

Fig. 8: Comparison of the average AUC and time among all methods in one diagram

we assess the trained method using image (a), (b), (c), and (d) as above. We observe negligible performance when the train and test data are from different sources. Details of the extensive experiments, their results, and the specifications of the train data are reported in the supporting document [30].

D. Detection Performance

We use the receiver operating characteristic (ROC) curve to evaluate the proposed strategy's experimental results. The area under the curve (AUC) is also calculated to express the ROC in a single measure. Larger AUC translates to higher ROC and better detection performance. Figs. 6 and 7 depict the ROC of the proposed method compared to other benchmark methods for the image (a), (b), respectively. Images (c) and (d) show similar results, which are presented in [30]; we discard depicting their ROC here to preserve brevity. Instead, we compare the average AUC versus time among the proposed and benchmark methods. Fig. 8 depicts this comparison, which clearly shows the superiority of the proposed method in terms of accuracy. Note that since the AUC average is close to its maximum value, all AUCs of the proposed method should also be close to one. However, such a conclusion can not be made for other methods. We also study the effect of window sizes on different performance measures in the supporting document [30]. There, we observe that ROC is independent of the window size.

Furthermore, Table I demonstrates the effect of various window sizes using AUC score and computation time for image (a), (b), (c) and (d), respectively. The best accuracy is emphasized with bolded text. Table II shows the AUC performance and the computation time for different detectors. According to Table II.(a), Our strategy's computation time is substantially superior to the CNND and BASO benchmark algorithms, which both are also supervised learning methods. Table II.(b) shows that the proposed method substantially outperforms the BASO and the CNND benchmark algorithms in terms of accuracy.

V. Conclusion

We proposed a deep learning detector for hyperspectral data anomaly detection. The main idea is based on training the network by reference data, labeling each test pixel with respect to its neighbors, and removing those objects whose size is not in the defined accepted range. It is imperative to choose reference data and test data from the same sensor because intrinsic sensor spectral features are among the network's features. The designed network can label each test pixel in less time compared to the state-of-the-art networks. It also has fewer learning parameters, resulting in significantly faster training. Finlay, extensive experimental evaluations reveal that the proposed approach substantially outperforms other state-of-the-art and traditional benchmarks.

References

[1] P. S. Thenkabail and J. G. Lyon, Eds., *Hyperspectral Remote Sensing of Vegetation*. CRC Press, apr 2016.

[2] L. Zhang, Q. Zhang, L. Zhang, D. Tao, X. Huang, and B. Du, "Ensemble manifold regularized sparse low-rank approximation for multiview feature embedding," *Pattern Recognition*, vol. 48, no. 10, pp. 3102–3112, oct 2015.

[3] D. Manolakis and G. Shaw, "Detection algorithms for hyperspectral imaging applications," *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 29–43, Jan 2002.

[4] C.-I. Chang and S.-S. Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE transactions on geoscience and remote sensing*, vol. 40, no. 6, pp. 1314–1325, 2002.

[5] C.-I. Chang, "Hyperspectral data processing: Algorithm design and analysis," *Hyperspectral Data Processing: Algorithm Design and Analysis*, 03 2013.

[6] Chein-I Chang and Shao-Shan Chiang, "Anomaly detection and classification for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 6, pp. 1314–1325, June 2002.

[7] D. W. J. Stein, S. G. Beaven, L. E. Hoff, E. M. Winter, A. P. Schaum, and A. D. Stocker, "Anomaly detection from hyperspectral imagery," *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 58–69, Jan 2002.

[8] I. S. Reed and X. Yu, "Adaptive multiple-band cfar detection of an optical pattern with unknown spectral distribution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 10, pp. 1760–1770, Oct 1990.

[9] J. M. Molero, E. M. Garzón, I. García, and A. Plaza, "Analysis and optimizations of global and local versions of the rx algorithm for anomaly detection in hyperspectral data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 2, pp. 801–814, April 2013.

[10] Heesung Kwon and N. M. Nasrabadi, "Kernel rx-algorithm: a nonlinear anomaly detector for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 2, pp. 388–397, Feb 2005.

[11] J. Zhou, C. Kwan, B. Ayhan, and M. T. Eismann, "A novel cluster kernel rx algorithm for anomaly and change detection using hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 11, pp. 6497–6504, Nov 2016.

[12] K. Tan, Z. Hou, D. Ma, Y. Chen, and Q. Du, "Anomaly detection in hyperspectral imagery based on low-rank representation incorporating a spatial constraint," *Remote Sensing*, vol. 11, no. 13, 2019.

[13] P. Horstrand, M. Diaz, R. Guerra, S. Lopez, and J. F. Lopez, "A novel hyperspectral anomaly detection algorithm for real-time applications with push-broom sensors," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, pp. 1–11, 2019.

[14] K. Tan, Z. Hou, F. Wu, Q. Du, and Y. Chen, "Anomaly detection for hyperspectral imagery based on the regularized subspace method and collaborative representation," *Remote Sensing*, vol. 11, no. 11, 2019.

[15] W. Li and Q. Du, "Collaborative representation for hyperspectral anomaly detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 3, pp. 1463–1474, March 2015.

[16] J. Li, H. Zhang, L. Zhang, and L. Ma, "Hyperspectral anomaly detection by the use of background joint sparse representation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2523–2533, June 2015.

[17] S. Chang, B. Du, and L. Zhang, "Baso: A background-anomaly component projection and separation optimized filter for anomaly detection in hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3747–3761, July 2018.

[18] X. Zhang, G. Wen, and W. Dai, "A tensor decomposition-based anomaly detection algorithm for hyperspectral image," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 5801–5820, Oct 2016.

[19] R. Zhao, B. Du, and L. Zhang, "Hyperspectral anomaly detection via a sparsity score estimation framework," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 6, pp. 3208–3222, June 2017.

[20] X. Kang, X. Zhang, S. Li, K. Li, J. Li, and J. A. Benediktsson, "Hyperspectral anomaly detection with attribute and edge-preserving filters," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5600–5611, Oct 2017.

[21] S. Li, K. Zhang, Q. Hao, P. Duan, and X. Kang, "Hyperspectral anomaly detection with multiscale attribute and edge-preserving filters," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 10, pp. 1605–1609, Oct 2018.

[22] A. Taghipour and H. Ghassemian, "Hyperspectral anomaly detection using attribute profiles," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 7, pp. 1136–1140, July 2017.

[23] J. Du and Z. Li, "A hyperspectral target detection framework with subtraction pixel pair features," *IEEE Access*, vol. 6, pp. 45 562–45 577, 2018.

[24] Y. Chen, Y. Wang, Y. Gu, X. He, P. Ghamisi, and X. Jia, "Deep learning ensemble for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 6, pp. 1882–1897, June 2019.

[25] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, pp. 1–12, 2015.

[26] W. Li, G. Wu, F. Zhang, and Q. Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 844–853, feb 2017.

[27] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, jul 2015.

[28] X. Kang, C. Li, S. Li, and H. Lin, "Classification of hyperspectral images by gabor filtering based deep network," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 4, pp. 1166–1178, April 2018.

[29] W. Li, G. Wu, and Q. Du, "Transferred deep learning for anomaly detection in hyperspectral imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 597–601, May 2017.

[30] M. Yousefan, H. E. Najafabadi, H. Amirkhani, H. Leung, and V. Hajihashemi, "Supplementary material for deep anomaly detection in hyperspectral images based on membership maps and object area filtering," 2020.

[31] G. Vane, M. Chrisp, H. Enmark, and S. Macenka, J., "Airborne Visible/Infrared Imaging Spectrometer (AVIRIS): An advanced tool for Earth remote sensing," in *From Res. Towards Operational Use*, vol. 2, Aug. 1984.

[32] K. Wu, E. Otoo, and K. Suzuki, "Optimizing two-pass connected-component labeling algorithms," *Pattern Analysis and Applications*, vol. 12, no. 2, pp. 117–135, Jun 2009.

[33] F. Verdoja and M. Grangetto, "Graph laplacian for image anomaly detection," *to be published, available at http://arxiv.org/abs/1802.09843v4*.